AMENDMENTS TO THE SPECIFICATION

Please amend the paragraph beginning on page 3, line 15, as follows:

In accordance with the present invention, a computer-readable medium containing a data structure called a base generator class is provided. An exemplary implementation of the base generator class comprises a base generator class constructor[[,]] and references a generator properties ~~object~~ class that provides incrementation capability, a status indicator, a schedule ~~object~~ class, and a logging ~~object~~ class.

Please amend the paragraph beginning on page 7, line 21, as follows:

The base generator class 302 also includes class properties 304, such as a generator description 308, status indicator 310[[,]]. The class properties 304 may also include references to a schedule ~~object~~ class 312, a logging ~~object~~ class 314, and a generator properties ~~object~~ class 316 that encompasses all the properties of [[the]] a generator, etc.

Please amend the paragraph beginning on page 7, line 26, as follows:

[[The]] An instance of the schedule ~~object~~ class 312 allows a user to specify when or after which condition a generator should start or end and a generator's recurrence pattern, if there is any. One exemplary implementation of the schedule ~~object~~ class 312 includes a start condition that needs to be fulfilled before object generation can occur. A start condition may be a specific time when a generator should be started; an amount of time that needs to pass after the generator has been invoked before the generator can be started; or an instantaneous start by the generator as soon as it is invoked. The execution of a generator may be scheduled to occur one time only, recur weekly or monthly, or recur after any given amount of time. The schedule ~~object~~ class 312 may also include an end condition that, once fulfilled, will indicate that no further occurrences of the schedule will take place. The ending of an execution of a generator may be at a specific time, after a certain amount of time, or after a given amount of objects. The schedule ~~object~~ class 312 may further include a dialog box that can be used to configure all schedule settings.

Please amend the paragraphs beginning on page 8, line 10, as follows:

The logging ~~object~~ <u>class</u> 314 enables the recording of what objects were generated, at what time, and using what properties. In one exemplary implementation of [[a]] <u>the</u> logging ~~object~~ <u>class</u> 314, structured query language (SQL) logging is used. For example, a database consisting of tables is used to store the logging information. One table is for each generator enabled for logging.

The logging functionality may be turned on or off by a user of a generator. The data resulted from using <u>an instance of</u> the logging ~~object~~ <u>class</u> 314 may be used by a verification process to verify what objects were generated with what generator settings. The logging ~~object~~ <u>class</u> 314 may also include a field that indicates whether a verification process has run on a generated object. In this way, the next time the verification process is run, it can skip the generated objects that are marked as having been verified.

As noted above, the base generator class 302 further contains a generator properties ~~object~~ <u>class</u> 316. [[The]] <u>An instance of the</u> generator properties ~~object~~ <u>class 302</u> manages all of the properties associated with the current generator. Each generator can have one or more properties that is specific to the type of work the generator performs. [[The]] <u>An instance of the</u> generator properties ~~object~~ <u>class</u> 316 manages the generator properties by keeping track of the number of properties defined for a generator, which generator owns the properties, adding or removing individual properties, etc.

In one embodiment of the present invention, the generator properties ~~object~~ <u>class</u> 316 ~~constitutes~~ <u>references</u> a generator property ~~object~~ <u>class</u> 317. [[The]] <u>An instance of the</u> generator property ~~object~~ <u>class</u> 317 represents an individual property. A generator property ~~object~~ <u>class</u> 317 may contain a corresponding property description that describes the purpose of [[the]] <u>a</u> property. A property may be defined as read-only, which means that a user of a generator cannot modify the property settings. The setting of a property may be cloned (copied) from one property into another property.

Please amend the paragraphs beginning on page 9, line 4, as follows:

[[The]] <u>An instance of the</u> generator property ~~object~~ <u>class</u> 317 also contains the value of a property. The value of a property may be a string value or a non-string value, which can be anything, such as a binary block of data or a reference to an instance of a class.

The generator property ~~object~~ <u>class</u> 317 may also contain properties and methods that enable a property value to vary from one generated object to the next. Because a generator may

22122AM.DOC

be used repeatedly to generate multiple objects, variation in property value may be needed on each object. The process to change a property value from one object to the next is called incrementation. For example, when a user uses a "Create File" generator to create 1000 files, the user may prefer to have the file names be differentiable. Incrementation is employed to change the value of property "FileName" from one file object to the next.

In one embodiment of incrementation of object generation, the value of a property is split into two parts--a stream portion and a numerical portion--the second portion is then incremented based on the incrementation settings and methods provided in [[a]] the corresponding instance of the generator property ~~object~~ class 317.

Please amend the paragraph beginning on page 10, line 21, as follows:

The generator property ~~object~~ class 317 further contains a default property incrementor method that increments strings, IP addresses, Media Access Control (MAC) addresses, and some other common values. A developer implementing a generator may implement a custom property incrementor for a generator property if the property value is to be incremented in some specific way. For example, a developer for a "Create File" generator may want to increment only the numeric portion of property value "file-1.txt." The developer hence needs to implement a custom property incrementor method to perform such an incrementation.

Please amend the paragraph beginning on page 11, line 1, as follows:

The generator property ~~object~~ class 317 also contains a validator method to validate the value of a property. A validator method may be used on a property regardless whether incrementation is supported on the property. By default, no validation is performed.

22122AM.DOC